AI planning and search emerged as its own field out of prior work on state-space search to suite the needs of applications such as robotics and scheduling problems. STRIPS (STanford Research Institute Problem Solver), written in LISP, was one of the first major projects in this field (Fikes and Nilsson, 1971). STRIPS was created to help control the famous robot Shakey at SRI (originally founded as the Stanford Research Institute, now known as SRI International), however it grew out of that original implementation into a more general purpose solution. STRIPS now primarily refers to the planning domain language that evolved out of this project, which introduced notions like actions having preconditions and postconditions, and is the primary inspiration for later planning languages.

Through the 1970's, planning for cases where there were subgoals was done by solving each subgoal separately and then concatenating the actions required to solve each subgoal. This was termed linear planning (Sacerdoti, 1975), however it was shown to be incomplete. In order to solve more generically, the plans to solve each subgoal may need to be interleaved, and one way of doing so is a technique called goal-regression planning (Waldinger, 1975). Goal-regression planning reorders actions so that the subgoals don't conflict with each other when possible to do so.

The majority of the 1980s and 1990s were devoted to partial-order planning, which minimizes instances of ordering actions in a plan. Some actions must take place after certain other ones, but the ordering of many actions may be interchangeable with each other within certain bounds. Compared to earlier linear planning, partial-order planning was more efficient, faster (Barrett and Weld, 1993) and could solve types of problems which stumped linear planning.

Graphplan (Blum and Furst, 1997) was another big step forward, delivering orders of magnitude improved speed compared to prior partial-order planning techniques. It is STRIPS-based, but instead of a traditional state-space, it creates a graph where each node is an action or fact, and the edges then connect these nodes such that action nodes connect to fact nodes which the action affects, and the fact nodes connect to action nodes for which they are conditions. While Graphplan performs well in problems that do not have too many objects, it has been shown to struggle when there are many objects (Helmert, 2001) and more traditional state-space search may still hold an edge in those cases.